In the Claims:

1. (previously presented) A method of implementing musical instrument digital interface (MIDI) synthesis comprising the steps of:

(a) providing a wireless handset having dual processor management control associated with a general purpose processor (GPP) and a digital signal processor (DSP), a flash memory having MIDI data stored therein, a digital-to-analog converter (DAC) and digital signal processor (DSP) peripherals to drive the DAC;

(b) interrogating the flash memory via the GPP to open a MIDI bit stream and determine sample sets to be loaded into a DSP memory associated with the DSP;

(c) loading and instantiating via the GPP, a DSP code associated with the sample sets into the DSP memory;

(d) initializing a sample set memory and signaling the DSP to start running a DSP synthesizer;

(e) parsing the MIDI bit stream into synthesis packets comprising MIDI commands via the GPP;

(f) transferring the synthesis packets to the DSP via the GPP; and

(g) time stamping and synthesizing the MIDI commands via the DSP to render audio signals to the DAC.

2. (previously presented) The method according to claim 1 further comprising the steps of:

(h) closing the MIDI bit stream when the MIDI bit stream has been exhausted;

(i) causing the DSP to stop synthesizing the MIDI commands; and

(j) de-allocating the sample set memory.

3. (previously presented) The method according to claim 1 further comprising the step of closing the MIDI bit stream in response to a user command.

4. (previously presented) A method of implementing musical instrument digital interface (MIDI) synthesis comprising the steps of:

(a) providing a wireless handset having dual processor management control associated with a first data processor and a second data processor, a flash memory having MIDI data stored therein, a digital-to-analog converter (DAC) and digital signal processor (DSP) peripherals to drive the DAC;

(b) interrogating the flash memory via the first data processor to open a MIDI bit stream and determine sample sets to be loaded into a shared memory;

(c) loading and instantiating via the first data processor, a second data processor code associated with the sample sets into the shared memory;

(d) initializing a sample set associated with the shared memory and signaling the second data processor to start running a MIDI synthesizer;

(e) parsing the MIDI bit stream into synthesis packets comprising MIDI commands via the first data processor;

(f) transferring the synthesis packets to the second data processor via the first data processor; and

(g) time stamping and synthesizing the MIDI commands via the second data processor to render audio signals to the DAC.

5. (previously presented) The method according to claim 4 further comprising the steps of:

(h) closing the MIDI bit stream when the MIDI bit stream has been completely read;

(i) causing the second data processor to stop synthesizing the MIDI commands; and

(j) de-allocating the sample set memory.

6. (currently amended) A musical instrument digital interface (MIDI) synthesizer comprising:

a flash memory having MIDI files stored therein;

a digital signal processor (DSP);

a digital-to-analog converter (DAC);

at least one DSP peripheral device operative to drive the DAC; and

a general purpose processor (GPP) for reading and parsing ~~configured to read and parse~~ the MIDI files stored in the flash memory to generate MIDI synthesizer commands therefrom, the DSP responsive to the MIDI synthesizer commands to synthesize audio signals and render the audio signals to the DAC via the at least one DSP peripheral device to implement a MIDI synthesizer.

7. (currently amended) ~~The~~ A musical instrument digital interface (MIDI) synthesizer comprising:

data storing means for storing MIDI files;

first data processing means for synthesizing audio signals;

data converting means for converting digital signals to analog signals;

driving means for driving the data converting means; and

second data processing means for reading and parsing the MIDI files stored in the data storing means to generate MIDI synthesizer commands therefrom, ~~wherein~~ the first data processing means ~~is~~ responsive to the MIDI synthesizer commands to synthesize audio signals and render the audio signals to the data converting means via the driving means to implement a MIDI synthesizer.

8. (previously presented) The MIDI synthesizer according to claim 7 wherein the data storing means comprises a flash memory.

9. (previously presented) The MIDI synthesizer according to claim 7 wherein the first data processing means comprises a digital signal processor and the second data processing means comprises a general purpose processor.

10. (previously presented) The MIDI synthesizer according to claim 7 wherein the first data processing means is word addressable and the second data processing means is byte addressable.